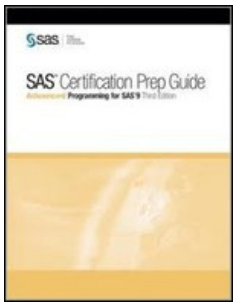


SAS Certification Prep Guide: Advanced Programming for SAS 9, Third Edition

by SAS Institute

Chapter 19: Introduction to Efficient SAS Programming

Overview

Introduction

As an experienced programmer, you want your SAS programs to obtain the desired results while minimizing the use of resources such as CPU time, real time, memory, and I/O. It is particularly important to optimize your SAS programs if you write or maintain production programs and work with large data sets. However, before you can select the most efficient programming technique to perform a particular task, you must carefully consider the technical environment and the resource constraints at your site. There is no single set of programming techniques that is most efficient in all situations. Instead, trade-offs in resource usage are associated with each technique.

In this chapter you will learn about analyzing the requirements for efficiency at your site and about running benchmarks to select the most efficient SAS programming techniques.

Note This chapter has no quiz.

Objectives

In this chapter, you learn to

- identify the resources that are used by a SAS program
- identify the main factors to consider when you assess the efficiency needs at your site
- identify common efficiency trade-offs
- select the appropriate SAS system option(s) to track and report the resource usage statistics that you want in your operating environment
- interpret resource usage statistics that are displayed in the SAS log in the z/OS, UNIX, and Windows operating environments
- identify the guidelines that you should follow in order to benchmark effectively.

Overview of Computing Resources

Running a SAS program requires the programmer and the computer to perform a variety of tasks. The *programmer* must determine the program specifications, write, submit, test, and maintain the program. The *computer* must load the required SAS software components and the program into memory, compile the program, locate the data that the program requires, execute the program, and store the SAS data set or output report.

The following resources are used to run a SAS program:

Resource	Description
 programmer time	the amount of time required for the programmer to write, test, and maintain the program. Programmer time is difficult to quantify, but it can be decreased through well-documented, logical programming practices and the use of reusable SAS code modules.
 CPU time	the amount of time the central processing unit (CPU) uses to perform requested tasks such as calculations, reading and writing data, conditional logic, and iterative logic.

real time 	the clock time (elapsed time) it takes to execute ajob or step. Real time is heavily dependent on the capacity of the system and on the load (the number of users that are sharing the system's resources). Because you cannot always control the capacity and the load demands on your system, real time is sometimes a less useful measure of program efficiency than CPU time. However, excessive use of real time often motivates programmers to improve a program's efficiency. Some procedures in SAS 9.1 give you the option of using threaded processing to reduce real time. Threaded processing can cause an increase in CPU time, so it is recommended that you track both CPU time and real time.
memory 	the size of the work area in volatile memory that is required for holding executable program modules, data, and buffers.
data storage space 	the amount of space on a disk or tape that is required for storing data. Data storage space is measured in a variety of units, some of which are used only in certain operating environments, as described below: ▪ All operating environments use bytes, kilobytes, megabytes, gigabytes, and terabytes. ▪ z/OS also uses blocks, tracks, and cylinders.
I/O 	a measurement of the read and write operations that are performed as data and programs are copied from a storage device to memory (input) or from memory to a storage or display device (output).

Assessing Efficiency Needs at Your Site

The first step in making an effective decision about how to optimize your SAS programs is to assess your site's technical environment, your program(s), and your data.

Assessing Your Technical Environment

To determine which resources are scarce or costly at your site, work with your IT department to analyze the following characteristics of your technical environment:

Category	Characteristics
hardware	▪ amount of available memory ▪ number of CPUs ▪ number and type of peripheral devices ▪ communications hardware ▪ network bandwidth ▪ storage capacity ▪ I/O bandwidth ▪ capacity to upgrade
operating environment	▪ resource allocation ▪ scheduling algorithms ▪ I/O methods
system load	▪ number of users or jobs sharing system resources

	▪ network traffic (expected) ▪ predicted increase in load in the future
SAS environment	▪ which SAS software products are installed ▪ number of CPUs and amount of memory allocated for SAS programming ▪ which methods are available for running SAS programs at your site

In most cases, one or two resources are the most limited or most expensive for your programs. You can usually decrease the amount of critical resources that are used if you are willing to sacrifice some efficiency of the resources that are less critical at your site.

Assessing Your Programs

Developing an efficient program requires time and thought. To determine whether the additional amount of resources saved is worth the time and effort spent to achieve the savings, consider the following characteristics of each of your programs:

Characteristics	Guidelines for Optimizing
size of the program	As the program increases in size, the potential for savings increases. Focus on improving the efficiency of *large programs*.
number of times the program will run	The difference in resources used by an inefficient program and an efficient program that is run once or a few times is relatively small, whereas the cumulative difference for a program that is run frequently is large. Focus on improving the efficiency of programs that are *run many times*.

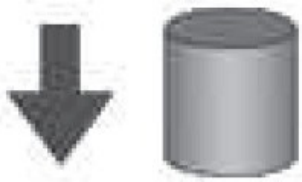
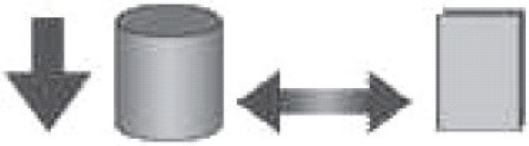
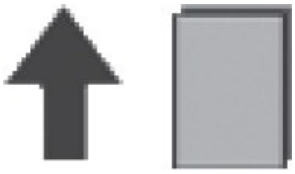
Assessing Your Data

The effectiveness of any efficiency technique depends greatly on the data with which you use it. When you know the characteristics of your data, you can select the techniques that take advantage of those characteristics. Consider the following characteristics of your data:

Characteristic	Guidelines for Optimizing
volume of data	As the volume of data increases, the potential for savings also increases. Focus on improving the efficiency of programs that use *large data sets or many data sets*.
type of data	Specific efficiency techniques might work better with some *types of data* (for example, data that has missing values) than with others.

Understanding Efficiency Trade-offs

When you are trying to optimize SAS programs, it is important to understand that there are trade-offs. Decreasing the use of one resource frequently increases the use of another. The following table shows examples of some common efficiency trade-offs.

Decreasing usage of this resource …	Might increase usage of this resource …
disk space 	CPU time 
I/O (by reading or writing more data at one time) 	memory 

real time (by enabling threading in SAS 9) 	 CPU time
any computer resource (by increasing program complexity) 	programmer time 

As these trade-offs illustrate, *there is no single best way to optimize a SAS program; it depends on your situation*. However, this chapter and the following chapters provide information that can help you determine which programming techniques are most efficient in your environment:

- "Controlling Memory Usage" on page 711
- "Controlling Data Storage Space" on page 730
- "Using Best Practices" on page 766
- "Selecting Efficient Sorting Strategies" on page 810
- "Querying Data Efficiently" on page 861

Using SAS System Options to Track Resources

You can specify one or more of the SAS system options *STIMER, MEMRPT, FULLSTIMER*, and *STATS* to track and report on resource utilization. The availability, usage, and functionality of these options vary by operating environment, as described below:

Option	z/OS	UNIX and Windows
STIMER	Specifies that the *CPU time* is to be *tracked* throughout the SAS session. Can be set at *invocation only*. Is the *default* setting.	Specifies that *CPU time and realtime statistics* are to be *tracked and written* to the SAS log throughout the SAS session. Can be set *either* at invocation or by using an OPTIONS statement. Is the *defaults* setting.
MEMRPT	Specifies that *memory usage statistics* are to be *tracked* throughout the SAS session. Can be set *either* at invocation or by using an OPTIONS statement. Is the *default* setting.	*Not available* as a separate option; this functionality is part of the FULLSTIMER option.
FULLSTIMER	Specifies that *all available resource usage statistics* are to be *tracked and written* to the SAS log throughout the SAS session. Can be set *either* at invocation or by using an OPTIONS statement. In the z/OS operating environment, FULLSTIMER is an alias for the FULLSTATS option. This option is ignored unless STIMER or MEMRPT is in effect.	Specifies that *all available resource usage statistics* are to be *tracked and written* to the SAS log through out the SAS session. Can be set *either* at invocation or by using an OPTIONS statement. In Windows operating environments, some statistics will not be calculated accurately unless FULLSTIMER is specified at invocation.
STATS	Tells SAS to *write* statistics that are tracked by any combination of the preceding options to the SAS log.	*Not available* as a separate option.

	Can be set *either* at invocation or by using an OPTIONS statement. Is the *default* setting.	

SAS system options are initialized with default settings when SAS is invoked. However, the default settings for some SAS system options vary both by operating environment and by site. For details, see the SAS documentation for your operating environment.

You can turn off any of these system options by using the options below:

- NOSTIMER
- NOMEMRPT
- NOFULLSTIMER
- NOSTATS.

Note In the z/OS operating environment, NOFULLSTIMER is an alias for the NOFULLSTATS option.

Note For information about using system options to track resource usage in your

operating environment, see the SAS documentation for your operating environment. Guidelines for interpreting the statistics that are generated by the FULLSTIMER SAS system option are also available at support.sas.com.

Note You can also use SAS Application Response Measurement (ARM) macros to monitor the performance of your applications. ARM macros are not covered in this course. To learn more about ARM macros, see the SAS documentation and detailed information about ARM macros at support.sas.com.

Using Benchmarks to Compare Techniques

To decide which SAS programming technique is most efficient for a particular task, you can *benchmark* (measure and compare) the resource usage for each technique that you are comparing. You should benchmark with the actual data to determine the most efficient technique.

Guidelines for Benchmarking

Your benchmarking is most likely to yield useful results if you follow these guidelines:

- *Before you test the programming techniques, turn on the SAS system options that report resource usage*.

  As explained earlier, to track and report on resource usage, you can use some or all of the system options STIMER, MEMRPT, FULLSTIMER, and STATS. The availability, usage, and functionality of these options vary by operating environment. You can also specify MSGLEVEL=I to display additional notes in the SAS log. Use the FULLSTIMER option to log a complete list of resources used.

  Note For more information about the SAS system option MSGLEVEL=, see "Creating Samples and Indexes" on page 470 or "Creating and Managing Indexes Using PROC SQL" on page 238.

- *Execute the code for each programming technique in a separate SAS session*.

  The first time that program code (including the DATA step, functions and formats, and SAS procedures) is referenced, the operating system might have to load the code into memory or assign virtual address space to it. The first time data is read, it is often loaded into a cache from which it can be retrieved more quickly the next time it is read. The resource usage that is required for performing these actions is called overhead. Using separate SAS sessions for each technique change can minimize the effect of the overhead on your resource statistics.

- *In each programming technique that you are testing, include only the SAS code that is essential for performing the task*.

  If you include too many elements in the code for each technique, you will not know what caused the results. If the program that you are benchmarking is not large, you can optimize it by changing individual programming techniques, one at a time, and running the entire program after each change to measure the effect on resource usage. However, a

more complex program might be easier to optimize by identifying the steps that use the most resources and extracting those steps into separate programs. You can measure the effects of different programming techniques by repeatedly changing, running, and measuring the separate programs. When isolating parts of your program, be careful to measure their resource usage under the conditions in which they are used in the complete program.

- *If your system is doing other work at the same time that you are running your benchmarking tests, be sure to run the code for each programming technique several times*.

  Running the code several times reduces any variability in resource consumption that is associated with other work that the system is doing. How you handle multiple measurements depends on the resource, as indicated below:

  - Use the minimum *real time* and *CPU time* measurements, because these represent most closely the amount of time your programming technique actually requires. The larger time values (especially in the case of real time) are the result of interference from other work that the computer was doing while your program ran.
  - The amount of *memory* should not vary from trial to trial. If memory does vary, it is possible that your program sometimes shares a resource with another program. In this situation, you must determine whether the higher or lower memory consumption is more likely to be the case when your program is used in production.
  - *I/O* can be an especially elusive resource to measure. With modern file systems and storage systems, the effect of your program on the I/O activity of the computer sometimes must be observed by operating system tools, file system tools, or storage system tools because it cannot be captured by your SAS session. Data is often aggressively cached by modern file systems and storage systems, and file caches are greatly affected by other activity in the file system. Be realistic when you measure I/O—it is possible to achieve good performance on a system that is not doing other work, but performance is likely to worsen when the application is deployed in a more realistic environment.

- *Run your benchmarking tests under the conditions in which your final program will run*.

  Results might vary under different conditions, so it is important to control the conditions under which your benchmarks are tested. For example, if batch execution and large data sets are used in your environment, you should incorporate these conditions into your benchmarking environment.

- *After testing is finished, consider turning off the options that report resource usage*.

  The options that report resource usage consume resources. If it is a higher priority in your environment to minimize resource usage than to periodically check an application's resource usage, then it is most efficient to turn off these options.

  Note To turn off the FULLSTIMER option, use the following statement:

  ```
  options nofullstimer;
  ```

Summary

Overview of Computing Resources

Resources that are required for running a SAS program include the following: programmer time, CPU time, real time, memory, data storage space, and I/O.

Assessing Efficiency Needs at Your Site

To make an effective decision about how to optimize your SAS programs, work with your IT department to assess the following factors at your site: the technical environment, your individual SAS programs, and the data.

Understanding Efficiency Trade-offs

It is important to understand the trade-offs that are involved in optimizing your SAS programs. Decreasing the use of one resource frequently increases the use of another. There is no single best way to optimize a SAS program; it depends on your situation.

Using SAS System Options to Track Resources

You can specify one or more of the SAS system options STIMER, MEMRPT, FULLSTIMER, and STATS to track and report on resource utilization. (In the z/OS environment, FULLSTIMER is an alias for FULLSTATS.) The availability, usage, and functionality of these options varies by operating environment.

Using Benchmarks to Compare Techniques

To determine which SAS programming technique is most efficient for a particular task, you can benchmark (measure and compare) the resource usage of each technique.